[Embedded Art - Project Group](https://www.facebook.com/embedded.art)  
*heart emoticon* oops interview Question *heart emoticon*  
1. What is difference between C and C++ ?  
*heart emoticon*

C++ is Multi-Paradigm ( not pure OOP, supports both procedural and object oriented) while C follows procedural style programming.  
.  
In C data security is less, but in C++ you can use modifiers for your class members to make it inaccessible from outside.  
.  
C follows top-down approach ( solution is created in step by step manner, like each step is processed into details as we proceed )   
.  
C++ follows a bottom-up approach ( where base elements are established first and are linked to make complex solutions ).

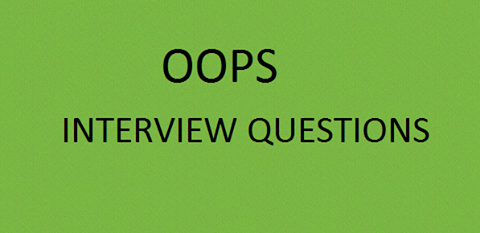
C++ supports function overloading while C does not support it.

C++ allows use of functions in structures, but C does not permit that.

C++ supports reference variables ( two variables can point to same memory location ). C does not support this.

C does not have a built in exception handling framework, though we can emulate it with other mechanism. C++ directly supports exception handling, which makes life of developer easy.  
.  
2. What is a class?  
*heart emoticon*  
Class defines a datatype, it's type definition of category of thing(s). But a class actually does not define the data, it just specifies the structure of data.   
To use them you need to create objects out of the class.   
Class can be considered as a blueprint of a building, you can not stay inside blueprint of building, you need to construct building(s) out of that plan.   
You can create any number of buildings from the blueprint, similarly you can create any number of objects from a class.

3. What is an Object/Instance?  
*heart emoticon*  
Object is the instance of a class, which is concrete. From the above example, we can create instance of class Vehicle as given below  
.............................Vehicle vehicleObject;...............................  
We can have different objects of the class Vehicle, for example we can have Vehicle objects with 2 tyres, 4tyres etc. Similarly different engine capacities as well.  
.  
4. What do you mean by C++ access specifiers ?  
*heart emoticon*  
Access specifiers are used to define how the members (functions and variables) can be accessed outside the class. There are three access specifiers defined which are public, private, and protected  
.  
private:  
Members declared as private are accessible only with in the same class and they cannot be accessed outside the class they are declared.  
.  
public:  
Members declared as public are accessible from any where.  
.  
protected:  
Members declared as protected can not be accessed from outside the class except a child class. This access specifier has significance in the context of inheritance.  
.  
5. What are the basics concepts of OOP?  
*heart emoticon*  
Encapsulation :  
Encapsulation is the mechanism by which data and associated operations/methods are bound together and thus hide the data from outside world. It's also called data hiding.  
In c++, encapsulation achieved using the access specifiers (private, public and protected). Data members will be declared as private (thus protecting from direct access from outside) and public methods will be provided to access these data.  
.  
Data abstraction :  
Data abstraction refers to hiding the internal implementations and show only the necessary details to the outside world. In C++ data abstraction is implemented using interfaces and abstract classes.  
.  
Inheritance :  
Inheritance allows one class to inherit properties of another class. In other words, inheritance allows one class to be defined in terms of another class.  
.  
6. What is the use of volatile keyword in c++? Give an example.  
*heart emoticon*  
Most of the times compilers will do optimization to the code to speed up the program. For example in the below code,  
*smile emoticon*  
int a = 10;  
while( a == 10){  
// Do something  
}  
*smile emoticon*  
compiler may think that value of 'a' is not getting changed from the program and replace it with 'while(true)', which will result in an infinite loop. In actual scenario the value of 'a' may be getting updated from outside of the program.  
.  
Volatile keyword is used to tell compiler that the variable declared using volatile may be used from outside the current scope so that compiler wont apply any optimization. This matters only in case of multi-threaded applications.  
.  
In the above example if variable 'a' was declared using volatile, compiler will not optimize it. In shot, value of the volatile variables will be read from the memory location directly.  
.  
7. In how many ways we can initialize an int variable in C++?  
*heart emoticon*  
In c++, variables can be initialized in two ways, the traditional C++ initialization using "=" operator and second using the constructor notation.  
Traditional C++ initilization int i = 10;  
.  
variable i will get initialized to 10.  
.  
Using C++ constructor notation  
int i(10);  
.  
8.What is implicit conversion/coercion in c++?  
*heart emoticon*  
Implicit conversions are performed when a type (say T) is used in a context where a compatible type (Say F) is expected so that the type T will be promoted to type F.  
short a = 2000 + 20;  
In the above example, variable a will get automatically promoted from short to int. This is called implicit conversion/coercion in c++.  
.  
9. What are C++ inline functions?  
*heart emoticon*  
C++ inline functions are special functions, for which the compiler replaces the function call with body/definition of function. Inline functions makes the program execute faster than the normal functions, since the overhead involved in saving current state to stack on the function call is avoided. By giving developer the control of making a function as inline, he can further optimize the code based on application logic. But actually, it's the compiler that decides whether to make a function inline or not regardless of it's declaration. Compiler may choose to make a non inline function inline and vice versa. Declaring a function as inline is in effect a request to the compiler to make it inline, which compiler may ignore. So, please note this point for the interview that, it is up-to the compiler to make a function inline or not.  
.  
10. What do you mean by translation unit in c++?  
*heart emoticon*  
We organize our C++ programs into different source files (.cpp, .cxx etc). When you consider a source file, at the preprocessing stage, some extra content may get added to the source code ( for example, the contents of header files included) and some content may get removed ( for example, the part of the code in the  of  block which resolve to false/0 based on the symbols defined). This effective content is called a translation unit. In other words, a translation unit consists of  
>Contents of source file  
>Plus contents of files included directly or indirectly  
>Minus source code lines ignored by any conditional pre processing directives ( the lines ignored by #ifdef,#ifndef etc).  
.  
11. What do you mean by internal linking and external linking in c++?  
*heart emoticon*  
[This interview question is related to questions on "translation unit" and "storage classes"]  
A symbol is said to be linked internally when it can be accessed only from with-in the scope of a single translation unit. By external linking a symbol can be accessed from other translation units as well. This linkage can be controlled by using static and extern keywords.  
.  
12. What do you mean by storage classes?  
*heart emoticon*  
Storage class are used to specify the visibility/scope and life time of symbols(functions and variables). That means, storage classes specify where all a variable or function can be accessed and till what time those variables will be available during the execution of program.  
.  
13. How many storage classes are available in C++?  
*heart emoticon*  
Storage class are used to specify the visibility/scope and life time of symbols(functions and variables). That means, storage classes specify where all a variable or function can be accessed and till what time those variables will be available during the execution of program.  
Following storage classes are available in C++  
Auto :  
It's the default storage class for local variables. They can be accessed only from with in the declaration scope. auto variables are allocated at the beginning of enclosing block and deallocated at the end of enclosing block.  
Register :  
It's similar to auto variables. Difference is that register variables might be stored on the processor register instead of RAM, that means the maximum size of register variable should be the size of CPU register ( like 16bit, 32bit or 64bit). This is normally used for frequently accessed variables like counters, to improve performance. But note that, declaring a variable as register does not mean that they will be stored in the register. It depends on the hardware and implementation.  
Static :  
A static variable will be kept in existence till the end of the program unlike creating and destroying each time they move into and out of the scope. This helps to maintain their value even if control goes out of the scope. When static is used with global variables, they will have internal linkage, that means it cannot be accessed by other source files. When static is used in case of a class member, it will be shared by all the objects of a class instead of creating separate copies for each object.  
Extern :  
Extern is used to tell compiler that the symbol is defined in another translation unit (or in a way, source files) and not in the current one. Which means the symbol is linked externally. extern symbols have static storage duration, that is accessible through out the life of program. Since no storage is allocated for extern variable as part of declaration, they cannot be initialized while declaring.  
Mutable :  
mutable storage class can be used only on non static non const data a member of a class. Mutable data member of a class can be modified even is it's part of an object which is declared as const.  
.  
14. What do you mean by persistent and non persistent objects?  
*heart emoticon*  
Persistent objects are the ones which we can be serialized and written to disk, or any other stream.   
So before stopping your application, you can serialize the object and on restart you can deserialize it. [ Drawing applications usually use serializations.]  
Objects that can not be serialized are called non persistent objects. [ Usually database objects are not serialized because connection and session will not be existing when you restart the application. ].  
.  
15. Is it possible to get the source code back from binary file?  
*heart emoticon*  
Technically it is possible to generate the source code from binary. It is called reverse engineering. There are lot of reverse engineering tools available. But, in actual case most of them will not re generate the exact source code back because many information will be lost due to compiler optimization and other interpretations.  
*heart emoticon* [See More](https://www.facebook.com/embedded.art)

[[](https://www.facebook.com/embedded.art/photos/a.605029662889104.1073741828.605015512890519/933399956718738/?type=1)](https://www.facebook.com/embedded.art/photos/a.605029662889104.1073741828.605015512890519/933399956718738/?type=1)